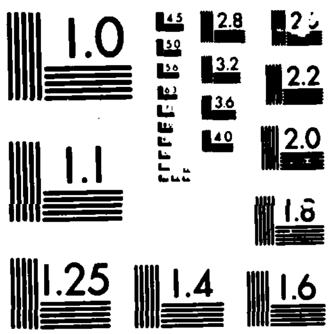
AD-A169 508 MODELING AND SIMULATION I INTRODUCTION AND GUIDELINES 1/1
(U) BROWN UNIV PROVIDENCE RI CENTER FOR NEURAL SCIENCE
J D DANIELS ET AL. 30 JUN 86 N00014-81-K-0136

UNCLASSIFIED F/G 6/16 NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

(12)

AD-A169 508

DTIC FILE COPY

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER #31	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Modeling and Simulation I: Introduction and Guidelines		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER
AUTHOR(s) J. D. Daniels and A. B. Saul		8. CONTRACT OR GRANT NUMBER(s) N000-14-K-0136 81
PERFORMING ORGANIZATION NAME AND ADDRESS Center for Neural Science Brown University Providence, Rhode Island 02912		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N-201-484
CONTROLLING OFFICE NAME AND ADDRESS Personnel and Training Research Program Office of Naval Research, Code 442 PT		12. REPORT DATE June 30, 1986
		13. NUMBER OF PAGES 26 pages
MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited. Publication in part or in whole is permitted for any purpose of the United States Government.

DTIC ELECTE JUL 7 1986 S D B

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Published in Journal of Electrophysiological Techniques, Vol. 13, pp.95-109, 1986.

Keywords: Digital [illegible]

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Modeling, Computer-aided engineering, Electrophysiology [illegible]
Simulation,
Neural Systems,
Differential Equation,

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

For neurophysiologists not normally engaged in modeling and simulation, we introduce the subjects. The availability of low-cost, high performance digital computer systems allows the development of detailed models, and their simulations, which can greatly aid the progress of experimentation. We review the ready-to-use simulation routines on IMSL Ins. libraries. Problems of error analysis in numerical methods are discussed. The virtues of general-purpose workstations for computer-aided-engineering are listed, and the application of such workstations to neural modeling is pointed out. This paper is the first [illegible]

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

86 7 7 066

first of two-the second provides a detailed look at a particular example of modeling: Plasticity in kitten visual cortex.

S N 0102- LF- 014- 6601

J.D. Daniels & A.B. Saul

Division of Engineering & Center for Neural Sciences

Box D

Brown University, Providence, Rhode Island 02912

Modeling and Simulation I: Introduction and Guidelines.

Key Words: modeling, simulation, nerual systems, differential equation, computer-aided engineering.

Send correspondence to J.D. Daniels.

Supported by ONR contract N00014-81-K-0136 and NEI grant EY04883. We acknowledge the scholarly help of Martha Motuz in the preparation of this manuscript. Three anonymous reviewers provided numerous helpful suggestions. Shawn K. Martion did excellent word processing with the bb editor and the troff formatter under UNIX.

Abstract. For neurophysiologists not normally engaged in modeling and simulation, we introduce the subjects. The availability of low-cost, high-performance digital computer systems allows the development of detailed models, and their simulations, which can greatly aid the progress of experimentation. We review the ready-to-use simulation routines on IMSL Inc. libraries. Problems of error analysis in numerical methods are discussed. The virtues of general-purpose workstations for computer-aided-engineering are listed, and the application of such workstations to neural modeling is pointed out. This paper is the first of two--the second provides a detailed look at a particular example of modelling: Plasticity in kitten visual cortex.

Introduction

A neurophysiologist takes pride in understanding how at least a small part of an animal's nervous system functions. The understanding ultimately is built upon knowledge gained by direct observation and experimentation with animals. In this and a subsequent paper, we address an important *adjunct* to direct experimentation -- expressing hypotheses as *quantitative relationships* (models) and *simulating* the models to verify or predict in detail the outcome of experimental tests.

This first paper discusses:

(1) When modeling and simulation are appropriate,
(2) How to begin the process of modeling and what features a good model will have,
(3) A guide to simulation techniques and error analysis,
(4) General-purpose layout and simulation packages for logic circuit design and their possible value in neurophysiological modeling and simulation.

The first paper is intended to interest experimentalists who normally do little or no modeling in connection with their research. In the second paper, we explain in detail a model for development of single neuron plasticity in the mammalian visual cortex. By focusing on development, our model naturally features the use of *differential equations* to describe *growth* processes. By considering the simultaneous responses of *many* neurons, our model requires the use of *matrix algebra* to describe the strengths of interconnections in a network. More experienced modelers may be engaged by the second paper; we hope they will not be slighted by our omission of other, classic and well-known neurophysiological models, from Hodgkin & Huxley's to Schwartz' and Grossberg's work.

Distinction between modeling and simulation. A *model* is a *representation* of something important by something convenient. For us, "convenient" will mean a mathematical relationship, presumably based on an underlying mechanism which explains the "important" results of experiments. In some cases, solution of a model's differential equations, for an initial condition and a definite input, will provide an ***analytical*** formula, which will show once a particular relationship between variables and parameters. The analytical solution is desirable, but the sometimes not feasible, or can be obtained only in limited contexts or with unwarranted assumptions. In such cases ***simulation*** helps. Simulation is an algorithmic ***search*** for the answer to a modeling problem, given a particular starting point and driving function. ***Numerical analysis*** embodies the set of general techniques for implementing the simulation, usually on a digital computer.

Relevance. Beyond their obvious usefulness in hypothesis testing, two other phenomena help make modeling and simulation relevant for all neurophysiologists:

(1) The continuing decline in the price of fast computing machines with large memories, friendly operating systems, and good graphic capabilities, make simulations of large or complex systems feasible to more researchers. We think ***complex*** models are best built with ***simple*** components (modules), however numerous and interconnected the components may be. The temptation to use computing power to simulate ***ad hoc*** arrangements of empirical relationships should be resisted. Later in this paper (Part 4) we will indicate that computer-aided-engineering (CAE) systems for digital circuit design offer features which help organize modeling around hierarchical modules.

(2) The growing demands of animal rights activists, and some politicians, to encourage researchers to substitute computer simulations for direct experimentation on mammals make simulation seem more valuable than it really is. Much emotional prose has been written about animal rights and unfriendly questions will continue to be asked about the necessity of animal

experimentation. It will behoove researchers involved in mammalian experimentation at least to be aware of the limitations and virtues of simulation, if for no other reason than to answer the inevitable questions. Researchers should appreciate, however, that demands often go beyond relatively straight-forward issues of animal care and surgical anesthesia; rather the matter for many seems to be *justification* for *any* use of mammals in research[37]. We can note that research done only by simulation is not science at all but an exercise in precision speculation. In such a case, the frustrations of researchers seeking to test their models on animals will be matched only by the hypocrisy of people who rally against the use of cats, dogs and monkeys in biomedical research while those same people are willing to accept medical diagnoses and treatments developed in animal experimentation research.

1. When should modeling be attempted?

Careful unbiased observation, often with the aid of elaborate equipment, generates raw data. Computer graphics can help in the *presentation* of data, allowing the researcher to view it from different perspectives. Statistical summaries can help compress the data presentations. The researcher spots a pattern and further organizes the data into relationships which accentuate the pattern. An underlying cause for the pattern is postulated. How precisely a cause-and-effect relationship can be formulated determines if the idea becomes a model. *If the research is at such a preliminary stage that major qualitative issues are undecided, then modeling is* premature[39]. Some researchers prefer to generate alternative qualitative hypotheses as a basis for further experiments. If *competition vs cooperation*, or *pump vs diffusion*, or *genetics vs environment* are the sorts of issues at stake, then perhaps simple yes-no experiments are all that are needed. However, *even when the focus is on choosing between alternative hypotheses,* *it is often possible to express each alternative quantitatively.* For example, suppose you postulate that motoneuron death during development is due to either a fixed genetic program or to lack of target tissue to innervate. Then experiments in which target tissue is decreased or increased from normal can help decide the issue, and show perhaps a *linear relationship* between amount of target tissue and number of surviving motoneurons[17].

Curve fitting. The process of modeling will generally result in one or more equations or graphs, to describe relationships between variables in the experiment. It is appropriate to use likelihood estimation techniques (often the least squares method) for finding the best values of *parameters* in the equations. There is a considerable literature on the subject of curve fitting (see Daniel & Wood[8], for example). The reader should be cautioned, however, about some of the more ambitious aims of curve fitting, beyond parameter estimation. In particular, it is possible to work with a set of input-output data by making the data fit a linear, logarithmic, or power relationship, or finding some polynomial to approximate it. We think that this latter use of curve fitting is to be avoided, because if a researcher must resort to devising model equations directly from the data, without a mechanism for supporting explanation, then the modeling process may again be *premature.* This is not to say that much literature on mathematical modeling does not emphasize such mechanism-independent exercises in problem solving, often for the good reason that some urgent need exists to *predict* the future course of some (usually economic) process -- see 1, 2, 3, 26, for example. Because modeling is applied to such a wide range of phenomena, from sociology to physics, some judgment--often common sense--is needed to choose approaches useful for neurophysiology research.

Statistical testing of hypotheses. None of the above remarks should be construed as discouragement from examining the *quality* of output data by use of various statistical tests, such as Chi-square. The reader should bear in mind, however, that, in general, such tests have little to do with *formulating* models, and that the hypotheses they test are often the computational result of generating simulation data from a true model. In the case of statistical analyses, computational formulas are necessarily *independent* of particular models. Of course, there are borderline cases. For example in deciding about the quantal nature of transmitter release from vesicles the *Poisson probability calculation* served as a *test* for the mechanism, not just as a means of analyzing the data[11].

Linear System Theory. This is a category of formalism which can tempt modelers of the CNS[38]. By collecting responses to step and sinusoidal inputs, then finding what combinations of linear differential elements can approximate the transient and steady state responses, researchers can fill in the "black box". Linear system theory emphasizes creating a set of first order differential equations to describe a process[12,35]. Once such a system is devised, the need is created to seek neural integrators, differentiators, filters and amplifiers in the neural tissue. To its credit, linear system theory has provided a number of useful models, especially in motor control, particularly eye movements[5,24]. On the whole, we view this black box approach as useful only if basic mechanisms suggest that reasonably linear elements may actually be part of a neuronal assembly.

Control of Stimuli. In some cases (for example, determining whether a certain *general environment* has a positive effect on development of cortical thickness[31]) the researcher has relatively little control over all the stimuli directly relevant to the output observed. Even precisely controllable sensory stimuli, such as visual patterns on a screen[10], may be several synapses away from the input fibers to neocortex the experimenter may really want to control. Those contemplating a model for a process should have at least some hope that the immediate inputs to the process can be known, and controlled. Otherwise, time may be better spent achieving such control, instead of inferring or guessing about the inputs to a model.

Modeling should be attempted when a mechanism (or *at least* a simple relationship) suggests itself as an explanation for trends in output data. Modeling is a way to move beyond linguistic labels for phenomena, beyond curve fitting of results, beyond black boxes, to an *understanding* and a way to *test* that understanding. Perhaps a model's highest calling is not to attempt premature explanations of too little data, but to synthesize new meaning from too much, conflicting, data[22]. Whether an attempt at modeling is successful may be decided by some of the issues addressed in the next section.

2. To Begin Modeling.

On the one hand, modeling can conjure visions of large-scale or small-scale *replicas* of the object under study. Even if mechanical or biochemical versions are replaced by electronic components, the resulting *hardware* model represents an extreme philosophy -- that the model should realistically duplicate as many features as possible of the object or process of interest.

On the other hand, the power of mathematical thinking to deal in abstractions can be carried to extreme. The process to be modeled can be considered an example of a general concept, requiring all aspects of the process to be converted to the formalism of the theory, often causing the original object to be greatly distorted. Linear system or sequential computer theory are often the distorting formalisms.

In neurophysiological modeling, it is perhaps unwise to drift to either extreme before more balanced approaches are worked with. To model the exact cable properties of every branch of every dendrite in a neuronal assembly, or to consider that some region of neo-cortex satisfies all the axioms of an algebraic vector field may be justified only if the input-output relationships of the relevant individual neurons have become well understood.

In this section we will consider the modest goal of starting a modeling project at one level, and in section 4 we will develop the notion of hierarchically modeling two or more levels. Here we first discuss **mathematical formulations,** especially differential equation forms; then we look at **fluctuations** in data, **mechanisms** for explanation, and end with a caution about anatomical **limitations** in modeling.

Models can be started in two ways, which are basically equivalent, and which should converge to the same final form. First consider the data from experiments. If variability is (at least temporarily) removed, some satisfaction can be gained by finding a simple mathematical function to describe the dependent and independent variable's relationship. We will hope that the mathematical function describes some physiological process or mechanism which generated the data in the first place. Second, consider starting with a list of such processes, and perhaps a "wiring diagram" which may sequence the action of the processes. A mathematical description of these processes, plus their arrangement in a wiring diagram, should produce a model similar to the data-driven one.

Unfortunately, either approach, especially the first, suffers from an embarrassment of riches. With regard to finding a function to represent a graphic or tabular or time series set of data it is a cliche to say that there are infinite possibilities. Even if there is only a finite set of orthogonal functions, the choice is still great. Suppose, for example, our output shows a damped oscillation as a function of time. We could choose a function which is a product of a decaying exponential, and a sinusoid (and gain factor G):

$$Y(t) = G * \exp(-t/a) * \sin(\omega t)$$

Or we could choose a *linear second order differential equation:*

$$a\frac{d^2y}{dt^2}+b\frac{dy}{dt}+c=f(\text{input, initial conditions})$$

What's the difference? Y(t) can be *a solution* to the equation. Generally, however, it is not explicit how the expression Y(t) depends on the particular input or initial conditions in the experiment. Presumably this could be dealt with by making G, a, and ω more complicated functions. On a more important level, the second order differential equation is appealing because it implies that (1) several different input-output curves, or separate experiments, contributed to the formulation, and (2) known rate relationships for underlying biological processes can simply be added together (linearity assumption).

Let us address the notion of *assumptions.* Critics of particular models search for unstated, *unrealistic, assumptions* the modeler may have made to achieve simplicity. But if if the goal is to model only one of the responses of a generally complicated neural tissue, simplifying assumptions made in setting up the model can be crucial to the generality of its success. There are three levels of assumptions:

(1) that the observed output is at least partly caused or controlled by a particular mechanism; (2) that a particular mechanism can be described by a certain equation; (3) that some terms in an equation may be inconsequential for *computational* purposes. The first two kinds of assumptions are certainly important in the beginning of the modeling process. The third kind of assumption is dealt with in section 3.

Our emphasis on differential (or difference) equation (DE) models has its own assumption: that we are modeling processes whose outputs (if not parameters) *vary as a function of time,* even given steady *inputs.* This is what we expect from processes, such as growth, decay, diffusion, and oscillation[4,36]. Expressing relationships in the form of DE's has more physical descriptive value, but has less of the immediate predictive value of a *formula* which is a *solution* to a DE. Eventually both the DE's and their solutions should be derived and generated, but the DE is more properly called a model and the solution is a particular answer.

Fluctuations. Besides uncertainty about underlying processes, biological modelers often have another problem not usually seen in physical science modeling: physical modeler-counterparts often do not: considerable variability in data. Much could be written about variability -- that one person's noise is another person's signal, etc., but it is inevitable that in *much* neurophysiological research, an experiment performed identically in two preparations can yields two different results. In beginning a modeling project, then, you must decide what to do about this variability. The simplest procedure is to *average* results over trials until noise is significantly *less than signal.* This has the disadvantage that you may end up ignoring time trends or functional subcategories. The specialty of *statistical modeling,* can be invoked to deal more carefully with *assumptions* about variability in results; Gilchrist[16] provides a good introduction. Essentially, you can make a decision to treat input or output as a *random variable.* What *distribution* your random variable takes then determines the subsequent statistical modeling. However, we advise that modelers be cautious about introducing random variables into their explanations, and instead make an effort to follow a *deterministic* pathway -- this should yield *simplicity,* in the beginning.

What mechanisms and basic laws can be called upon to model neural processes? Ideally, for reductionists at any rate, everything biological could be explained by fundamental physics and chemistry--there would be no need even to resort to membrane or synapse or nerve cell models as *basic* assumptions. In the absence of this ideal, consider four approaches:

Guidelines for Modeling

(1) Analogy. Assume the process to be modeled is like some well-defined mechanical or electrical element, like a spring or a capacitor.

(2) Function-from-structure. Usually the *physiological or behavioral* process can be recorded only at a level less microscopic than the underlying anatomical structure, and one can devise models, using those substructures, which account for the grosser responses measured in the experiment. We will see this approach in the second paper, when assumptions about *synaptic* function are called upon to explain responses of whole nerve cells.

(3) Physiological sub-units. Assume all the important *inputs* or projections to a certain process can be accounted for. If the physiological operations of these inputs in response to the relevant stimuli can be measured and quantified, then these operations can become the mechanisms for the model. This last procedure provides the most hope that the model can be *tested* without devising new, more microscopic methods of analysis. We will see it at work also in the second paper.

(4) Bottom-up direction. *Some* neural processes have been well characterized, and are now textbook examples--generation and propagation of action potential, release of transmitter, characteristics of channels, postsynaptic action of certain transmitters (including potentiation and habituation) etc. By restricting sub-units of the model to these relatively well understood mechanisms, the foundation of assumptions becomes more solid.

Further considerations about anatomical knowledge in neural modeling. In the experiment considered especially in the second paper, a nerve cell or group of cells responds to a stimulus, and that response serves as the object of our modeling effort. Being able to interpret the results of such an experiment depends on knowing *something* about the input to the cells in question. Ideally we would like to know *every* cell projecting to our target: where their synapses land in the dendritic tree, whether synapses are excitatory or inhibitory, and whether the target cell has any *reciprocal* connections with its input. In developmental modeling, the job may be to predict changes over time in the strength and arrangement of these connections. A modeler may encounter frustration if such basic anatomical knowledge is lacking. For examples, an annoyance of working with invertebrate CNS neurons is that connections must often be inferred from two-electrode stimulate-and-record experiments instead of direct anatomical tracing--small caliber fibers tangle together in dense neuropils which challenge the skill of neuroanatomists to unravel[20].

Once a wiring diagram is reasonably in hand, one can begin its description by labeling *nodes* and *branches*. The nodes can index the rows and columns of a square matrix, and the branches can fill in the magnitudes and signs of the matrix elements. For example, if node 3 connects with strength 4 to node 6, element $M_{36} = 4$. For further details, see any text on linear algebra, such as Hoffman & Kunze.

Other help with modeling. Besides the books on case studies 1, 2, 3, 26 cited in Section 1, there is a modest literature on *basic principles* of modeling, for biological and other fields. We have found Saaty & Alexander[33] and Cross & Moscardini[7] worthwhile introductions. Note that, especially if a new technique is involved, experiments will seem to generate new data which require a fresh approach to their understanding. However, the best *first* step in starting your model may be not to work with your own results, but to review what others have done in analogous situations. While this may seem mundane advice, and even frustrating, considering how fragmented the neural modeling literature is, it should be worthwhile. Additionally, once your model has been mathematically formulated you may find that other, different systems have been similarly structured, and solution techniques in those other systems may be useful to you.

What to hope for as the modeling proceeds. If mechanisms or subunits (primitives) have been identified, and an approach selected to model them, then the primitives can be linked together, with structural (anatomical) considerations. The result will be a set of *variables* and *parameters*. Values of the parameters need to be estimated. With even modest success constructing a tentative model, one becomes ready either for proving theorems about the model's solutions, stability, etc., or for *simulating* its performance. The modeler should keep in mind the relevance of his work to experimentation. As MacGregor & Lewis[25] say, "...the modeler who does not spell out how his model can be *contradicted* experimentally has not lived up to his responsibility..." (page 391).

Let us conclude this section with an example illustrating **limitations** in modelling. Consider a problem in neural development, but not one having to do with single unit recording: how to explain the *migration* of neural crest cells, in the embryo. A considerable amount of information is known about the timing and movement of newly generated cells, and their destinations in various parts of the emerging peripheral nervous system and other sites[23]. Even congenital deformities of the face, due to improper development of the neural crest, are known.

What we would like to know, in order to *model* this activity, is a sequential (and perhaps recursive) algorithm which instructs the migrating cells. In fact, progress has been made in a number of regards--structural cues, such as arrangement of glial cells, are known, and chemical signals, including cell adhesion molecules, seem to have influence. Yet, this information does not seem to be sufficient to suggest any particular growth algorithm. Important qualitative issues remain unsolved: Is the movement active or passive? What role does genetics play, independent of environment? What are the signals for *stopping* movement? Partly the problem is that simulation of the model would have to generate a three dimensional pattern of cell movement, a difficult enough problem in computer graphics, where an *instruction set* is provided, and a virtually hopeless problem when, in the neural crest case, the instruction set can only be guessed at. Another limitation for neural modellers is that migrating crest cells have not formed a *synaptic* information network, and the understandings from dealing with such a network are not available.

Basically, attempts to quantitatively model neural crest cell migration would be premature[39]. The best research direction is toward further *observations* of migration, while systematically manipulating the environment in which the migration takes place. *In vitro* systems which mimic the embryo *in vivo* would, according to LeDourain[23], provide the necessary breakthrough.

3. A Guide to Simulation Techniques and Error Analysis

Modeling by itself is worthwhile because the effort to model will require a researcher to confront the data and devise mechanistic explanations. However, the value of a model will increase when it is used to determine results from hypothetical inputs. If the model's output can be easily calculated by hand, so much the better; but often the model's performance must be *simulated*. How much effort should the researcher devote to *methods* of simulation? We think the effort should be minimal. This means, in practice, that almost anyone can avoid excessive programming by *using general purpose simulation packages*. In this section we are not going to review the simulation techniques themselves -- rather we will offer a brief *guide* to software available on various levels of hardware. It should be noted that some of the software - especially at the workstation level - is user-friendly: once a model has been laid out and an input specified, the computer will do *everything* required to give a graphical output without the user having to specify or understand the simulation technique.

The most succinct advice we can offer about simulation is this: Nearly all mainframe systems in scientific establishments have available the IMSL Inc. library of routines. The documentation for IMSL should be investigated as a first step[1*]. The main IMSL Library has over 500 Fortran subroutines, organized into three categories:

[1] *This is not meant to sound like an uncritical advertisement for IMSL, use of their routines has its own limitations. However, IMSL stands alone as the *main commercializer* of this special software

(1) Mathematics
- Interpolation, approximation & smoothing
- Differential equation solvers
- Eigensystem analysis
- Linear Algebraic equations--solutions
- Linear Programming
- Non-linear equations
- Optimization
- Transforms--FFT and inverse Laplace
- Vector & Matrix arithmetic

(2) Statistics
- Basic statistics--estimates of mean & variance, correlation, etc.
- Analysis of variance
- Categorized data analysis--contingency tables, life tables
- Non-parametric statistics
- Multivariate statistics, including maximum likelihood estimation
- Regression analysis
- Sampling (random) with inferences regarding mean
- Time series: time and frequency domain

(3) General Applications
- Generation and testing of random numbers
- Special functions--Gamma, Bessel, elliptical integrals, etc.
- Utility routines--sorting, printing etc.

The full library can be used on machines as small as Apollo 320's and DEC PDP-11's which support Fortran. All the user need worry about is the presence of a *standard* Fortran compiler on the machine[2*].

Once routines have been selected to simulate a model, a user needs to choose how the *input* will be presented to the simulator. In some cases the input may be created internally by a random number generator. Morgan[27] has useful advice about this statistical technique, including an introduction to Monte Carlo methods. In other cases, if the modeler wishes to use

[2] *IMSL has selected two subsets of frequently used subroutines for the IBM PC One, called MATH/PC, has subroutines from all of the categories of section (1) of the table above, except for linear programming, plus some special functions and utility routines The other, called STAT/PC, has a selection from each of the section (2) categories, plus some random number and utility routines

real data from an experiment, there will be need for a *communications interface* between the data-collecting apparatus and the computer for simulation.

Error Analysis. Using pre-packaged routines for numerical analysis removes the user from the algorithms and code underpinning the routines. Ideally the commercial packages will be well-tested modules. However, there is the drawback that difficult-to-detect *errors* may arise in the simulation process. Here we don't mean errors which can be corrected by debugging code, but errors *intrinsic to the use of a digital computer to approximate functions*[3*].

Numerical analysis errors are of three sorts:

(1) **data representation errors:** roundoff, chopping, base conversion, floating point.

(2) **functional approximation errors:** transcendental functions must be approximated by operations involving only addition and multiplication, or by table look-up. In either case, the approximation itself, even if perfect data is used, will result in some error.

(3) **truncation errors:** continuous processes, especially differential equations, are approximated by discrete *difference* methods. Truncation errors are expressed as the lowest power of the step size neglected by the numerical scheme. For example, the differential equation $y' = f(x,y)$ can be approximated by the Euler method as

$$y_{n+1} = y_n + hf(x_n, y_n),$$

which has a truncation error of order h^2, since the Taylor series

$$y(x+h) = y(x) + hy'(x) + \frac{h^2}{2} y''(x) + \ldots$$

becomes

$$y_{n+1} = y_n + hf(x_n, y_n) + \frac{h^2}{2} f'(x_n, y_n) + \ldots$$

upon substitution of

$x_{n+1} = x_n + h,\ y_n = y(x_n)$ and $y' = f(x,y)$.

The Euler method simply neglects all but the first two terms, so (assuming f is smooth) the error between the approximate and the exact solutions falls off quadratically as the step size h is decreased.

The subtraction of two nearly equal numbers is an operation notorious for introducing error. Because subtraction is used in it, numerical differentiation is sensitive to rounding errors and more difficult to work with than numerical integration.

In a limited way, the *accuracy* of a numerical routine can be improved at a sacrifice in *speed* of execution. Representing numbers by "double precision" is the start of this more accurate approach; giving the pre-packaged routines smaller stopping criteria[4*] continues it. If a numerical method is of order h^5 (such as the Runge-Kutta *method), halving the step size of the* search gives a 32-fold reduction in the truncation error at a cost of twice the run-time. However, the routines themselves often give you only limited information about the build-up of errors in computation. **There is no substitute for learning at least a little about errors in numerical analysis.** We have found Vandergraft[41] particularly helpful. Also, references 3, 6, 13, 28, 40 are worthwhile.

When one considers that a model is an approximation to a real physical situation, and that a simulation is an approximation to a model, a bit of cautious humility will be instilled. Beyond the textbook methods of calculating accumulated error in numerical techniques, the

[3] •If you try to improve your control over these sources of error by writing your own code, you are likely to use less optimal techniques than those in the commercially available libraries, and so be no better off

[4] •This is the size of the difference between computed and desired values at which an approximation procedure is stopped The smaller the stopping criteria size, the more steps required to achieve the approximation

modeler of experimental data has another, empirical, method of investigating the model's simulation sensitivity. The response of the simulation to small changes in inputs (whose outputs are known) can be examined. If the outputs do not fluctuate more than expected, then the modeler may have confidence that a "stable" simulation has been found.

AI techniques for simulation, and modeling. It is often claimed that human thought processes, and, presumably, animal nervous systems in general, have less in common with *computing* machines, than with fuzzy logic[43] *thinking* machines. To model *and* simulate such *cognitive* systems, researchers in artificial intelligence (AI) use the LISP language. A growing number of companies (Symbolics, Lisp Machines Inc., Texas Instruments, Xerox and Tektronix now in the market) are selling workstations specialized for LISP programming. We consider that, for the neural modeling discussed in this series of papers, such machines, at present, have *only limited* value. They are relatively weak in computing ability, compared to their capabilities in manipulating strings of linguistic labels. We think that by the end of the decade, though, such capabilities will become more important to neural modelers. Even now, isolated aspects of the modeling and simulation process can be help by AI-type programs. Witness SMP (Symbol Manipulation Program) created by Stephen Wolfram[42] and marketed by Inference Corp. SMP helps the modeler seek *analytical* solutions by dealing directly with *variables* in expressions. SMP factors, integrates, solves linear and nonlinear equations, and works with complex variables and matrices. It is available for VAX and Apollo installations. SMP should, in some circumstances, give modelers the freedom to combine together various expressions for primitives together in large awkward expressions, then use SMP to find a reduced form.

Analog Computers. In this paper, and in the next, we emphasize the use of *differential equations* to model neural processes of development. It may appear somewhat inefficient, then, to call for the use of digital computers in the simulation of such models, since numerical techniques on digital computers require approximating methods such as the trapezoid rule and difference equations for integration and differentiation. Why not use analog computers, where the proper placement of capacitors and resistors around op amps *automatically* creates elements which are almost exact counterparts of differentiators and integrators? Once the circuit is assembled, it will simulate without any need for programming. However appealing they may appear at first glance, we do not recommend simulation with analog computers, for the following reasons:

(1) Worse than programming, an analog computer designed to mimic a particular differential equation requires manual wiring of components and connections. Every *change* requires re-wiring. The physical routing of wires may be important, to avoid unwanted oscillations in amplifiers.

(2) The accuracy of the simulation depends on the tolerance range of the components, and the gain-bandwidth and noise characteristics of the amplifiers used. Such attributes may vary with temperature and age. Components and amplifiers with improved characteristics cost more money.

(3) Output is simply the time-varying voltages on selected op-amps--no graphic capability is normally present to display data in convenient ways. Even providing input may present problems. If input signals with random or noisy components are desired, special generators may have to be hooked into the system.

Basically, consideration of an analog computer's shortcomings, even with a natural system like differential equations, reminds us that analog computers are not "general purpose". They cannot be easily re-programmed to imitate a wide variety of systems, and therein lies the usefulness of a high-speed digital computer, whether in simulation or any other task.

In the next section we consider programs and supporting hardware designed to mimic digital (and linear) elements and circuits, and note how these computer-aided-engineering workstations may provide value to neural modelers.

4. Computer-aided layout and simulation packages for circuit design: Their potential applications to neural modeling.

Even though we have encouraged the use of differential equations to model neural processes, we have noted a tension between the equations' forms and the numerical techniques employed by digital computers to simulate them. To eliminate some of this tension, consider describing neural mechanisms in terms of computer logic elements -- counters, inverters, etc. If this is done, then we may be able to use some helpful modeling and simulation systems, namely today's generation of computer-aided-engineering (CAE) workstations for gate level logic design. We list features of these workstations:

(1) The modeler does not need to learn a programming language in order to use a CAE system. It's hardly even necessary to use a *keyboard* --A mouse or puck beside the terminal can be used to move a pointer on the screen. The user points to a command on a screen menu, then presses a button on the puck to execute the command.

(2) Unless an idiosyncratic part is required, the modeler does not have to *define* the functions of basic gates. These and other elements are stored in *data base libraries* accessible to the user through more menus to be called up on the screen.

(3) Computer techniques for drawing in elements and connecting them to each other have been refined by commercial development: First a menu of library components is placed on one screen window. The modeler moves the pointer to a desired component name and presses a puck button to highlight that component. Next the modeler moves the pointer to another window, to be used for drawing the model. In this window, when the user presses the puck button, a *symbol* of the previously-selected component will pop onto the screen. More components can be placed in this way. Beyond component *placement*, input and output pins on the components can be connected, in a *drawing* mode. A puck button is pressed at the *start* of a connection, and a different button is pressed when the pointer is at the *end*. As soon as the *end* button is pressed, a wire is routed between start and end points.

(4) Once a model drawing in the window is finished, it can be saved by passing the drawing through a *compiler*, which will generate a data-base description. If any primary wiring errors have been made, such as connecting two outputs together, the compiler can return error messages to help the user affect repairs.

(5) The description of the model, now stored in a data-base file, can be used as input to a *simulator*, which has access to algorithms defining the action of each primitive. To drive the simulator the user defines input waveforms and tells the simulator to what level of timing resolution output waveforms should be displayed.

(6) Simulators are given features of hardware logic analyzers -- breakpoints can be set, many waveforms viewed at once, with state or timing output format.

(7) If modeling and simulation are finished for one level of process, say a complete nerve cell, then the user can declare the nerve cell to be a ***new primitive*** and can begin using the new primitive at one level higher in a hierarchy. On the other hand, if, to finish the nerve cell model, a special "axon component" is needed, the user can descend to a lower level of hierarchy, construct and define the new component, and return to the original level to use it.

Simulator features. What distinguishes the use of computers for digital circuit design in the 1980's from the previous decade is the *simulator*. Previously, computers had helped manufacturers layout PC boards, generate net lists for wire wrapping machines, and perform other "anatomical" tasks. Finally software engineers began solving simulation problems on a large enough scale to be useful to designers of VLSI chips, where thousands of gates can be involved. To be effective, the simulators had to have internal models which take account of gate delay, fanout, and other characteristics of real logic gates. In some cases, where complex chips cannot be mathematically modeled, examples of actual chips could be *plugged into sockets in the simulator hardware* and accessed when their parts were required in the larger simulation. Speed is an important factor for any simulator, and digital logic simulators have helped push the state of the art in hardware accelerators. Simulator attachments which can perform over 100 million instructions per second are not uncommon now. It should be noted, however, that even with their speed, simulators hardly approaches real time, and waiting for the output of a simulator, or even a circuit diagram compiler, can take several minutes.

Analog simulation. The state of the art in making digital simulations of analog computers is not nearly as advanced as *digital* circuit simulation. Most companies offer SPICE, a program originating from UC Berkeley; SPICE is good for *designing* op amps from basic transistor elements, but not so useful for putting op amps in analog designs, as neural modelers might want to do. SPICE is not well integrated with the digital part of CAD (Computer-Aided Design) systems, nor with sophisticated graphic outputs.

The Marketplace. Three start-up companies were mainly responsible for creating the commercial interest in simulator-based CAE for circuit design--Daisy, Mentor and Valid (DMV). The three still have about 80% of the market. We have surveyed products from about 15 other small firms offering "second generation" systems. The main competition for DMV, however, will come from large companies like IBM, Hewlett Packard and Tektronix, who realize that if engineers can layout and simulate circuits inside computers, then the market for traditional test equipment (like oscilloscopes and function generators) will have a limited future.

A system similar to what we described in this section is likely to cost about S 50,000. The hardware can be either custom built, like Valid's S-32 or Daisy's Logician, or it can be a standard workstation, like the Apollos that Mentor uses for its Id`a 1000 software. All these systems are capable of storing schematics for and simulating systems of several thousand logic gates. If a modeler had more modest needs one of the CAE systems which can reside on an IBM PC might seem attractive. However in the tests we have given such systems (from p-CAD, FutureNet and Chancellor), they had frustratingly fewer features than the DMV types; we found them awkward to use.

Because the market for digital circuit CAD systems is growing (toward S1 billion per year), new machines, with more features, are regularly introduced. What may be lacking this year, for example in op amp and interface modeling, may be standard on next year's models. One way to keep up with this field is to read **VLSI Design** magazine, which has become a forum in the CAE field, for both ads and technical papers.

Summary

In this paper, we

(1) distinguish between modeling and simulation,

(2) offer general advice about when and how to model a process, and caution about simply curve-fitting data to find quantitative relationships,

(3) suggest the use of general purpose *simulation* libraries, such as IMSL, and discuss the problem of *error analysis* in numerical methods, perhaps the only problem the researcher need worry about in a fundamental way, once the process of simulation has begun,

(4) describe the features of computer workstation systems for laying out and simulating large scale digital circuit designs, and point out the usefulness of these systems to nerve cell modeling,

How will you know your exercise in modeling and simulation has been successful? There are two stages:

(1) Compare the results from experiments already performed, to the output of simulations intended to mimic those experiments. If there are discrepancies, adjust *parameters* in the model, or reformulate the model itself.

(2) Once classic results can be accounted for, use untested inputs to predict new results. If simulation is efficient, we may be able to test a great many novel inputs, perhaps using statistical techniques, to *search* for particularly important forms of output. For example, in the case to be analyzed in the second paper, we can vary the duration of input (visual experience) until we find a minimum time before which no plasticity (ocular dominance shift) occurs.

What are the limitations to success in modeling and simulation? We have already provided a number of cautions in the paper:

(1) There should be sufficient data to work with;

(2) Variability should be accounted for;

(3) Mechanisms should suggest themselves;

(4) The wiring diagram should be understood;

(5) The simulation should be examined for representation, calculation and truncation errors; it should not be too time-consuming for the available computer.

The reader should be reminded again that a model's job is to account for a small part of a nervous system's behavior. By simplification, it will do this in an approximate way. Likewise the simulation will approximate the model itself. If, after these restrictions from full reality, the model can reasonably predict behavior, it will be successful. Finally, we paraphrase Feynman' s[15] caution that, once formulated, equations can take on a life of their own. A good modeler will not become enamc ured of a particular formalism, and will thus avoid excessive frustration when the time comes inevitably to revise the model.

Appendix: Software sources.

(1) IMSL Inc., NBC Building, 7500 Bellaire Boulevard, Houston, TX 77036, 713-772-1927

(2) Inference Corp., Computer Mathematics Group, 5300 W Century Boulevard, Los Angeles, CA 90045, 213-417-7997

References

1. Boyce, W. M., ed. *Case Studies in Mathematical Modeling.* Boston: Pitman Publishing; 1981.

2. Bradley, R.; Gibson, R. D.; M. Cross, eds. *Case Studies in Mathematical Modeling.* New York: John Wiley & Sons; 1981.

3. Burden, R. L.; Faires, J. D.; Reynolds, A. C. *Numerical Analysis, 2nd Edition.* Boston: PWS Publishers; 1978.

4. Burghes, D. N.; Wood, A. D. *Mathematical Models in the Social, Management and Life Sciences.* New York: John Wiley & Sons; 1980.

5. Carpenter, R. H. S. *Movements of the Eyes.* London: Pion Limited; 1977.

6. Conte, S. D.; de Boor, C. *Elementary Numeric Analysis: An Algorithmic Approach.* New York: McGraw Hill; 1980.

7. Cross, M.; Moscardini, A. O. *Learning the Art of Mathematical Modeling.* New York: John Wiley & Sons; 1985.

8. Daniel, C.; Wood, F. S. *Fitting Equations to Data.* New York: John Wiley & Sons; 1971.

9. Daniels, J. D. M-Cells: A logic circuit model to account for some features of CNS inhibition. *Biol. Cybern. 29*: 1-9; 1978.

10. Daugman, J. Operation and interface manual--Picasso CRT image sythesizer. Innisfree Inc., P.O. Box 160, Cambridge, MA 02139; 1984.

11. Del Castillo, J.; Katz, B. Quantal components of the endplate potential. *J. Physiol. London,* 124: 560-573; 1954.

12. Director, S. W.; Rohrer, R. A. *Introduction to System Theory.* New York: McGraw-Hill; 1972.

13. Dorn, W. S.; McCracken, D. D. *Numerical Methods with FORTRAN IV Case Studies.* New York: Wiley; 1972.

14. Ermentrout, G. B.,; Cowan, J. D. A mathematical theory of visual hallucination patterns. *Biol. Cybern. 34*: 137-150; 1979.

15. Feynman, R. *The Character of Physical Law.* Cambridge, MA: The M.I.T. Press; 1965.

16. Gilchrist, W. *Statistical Modeling.* New York: John Wiley & Sons; 1984.

17. Hamburger, V. The effects of bud wing extirpation on the development of the central nervous system in chick embryos. *J. Exp. Zool. 68*: 449-494; 1934.

18. Holden, A. V., Winlow, W.; Haydon, P. G. The induction of periodic and chaotic activity in a molluscan neuron. *Biol. Cybern. 43*: 169-173; 1982.

19. Hurlbert, A.; Poggio, T. Spotlight on attention. *Trends in Neuroscience 8*: 309-311; 1985.

20. Kandel, E. R. *Cellular Basis of Behavior: An Introduction to Behavioral Neurobiology.* San Francisco, CA: W. H. Freeman and Company; 1976.

21. Keener, J. P. Chaotic Cardiac Dynamics. In *Mathematical Aspects of Physiology.* Edited by F. C. Hoppensteadt, Am. Math. Soc., Providence, RI, pages 299-325; 1981.

22. Kuhn, T. *The Structure of Scientific Revolutions, 2nd Edition.* Chicago: University of Chicago Press; 1970.

23. LeDouarin, N. *The Neural Crest.* New York: Cambridge Univ. Press; 1982.

24. Lennerstrand, G., Zee, D. S.; Keller, E. L., eds. *Functional Basis of Ocular Motility Disorders.* New York: Pergamon Press, 1982.

25. MacGregor, R. J.; Lewis, E. R. *Neural Modeling: Electrical Signal Processing in the Nervous System.* New York: Plenum Press; 1977.

26. Marcotorchino, J. F.; Proth, J. M.; Janssen, J. *Data Analysis in Real Life Environment: Ins and Outs of Solving Problems.* Amsterdam: Elsevier; 1985.

27. Morgan, B. J. T. *Elements of Simulation.* London: Chapman and Hall; 1984.

28. Ortega, J. M. *Numerical Analysis: A Second Course*, New York: Academic Press; 1971.

30. Regan, T.; Singer, P., eds. *Animal Rights and Human Obligations.* Englewood Cliffs, NJ: Prentice-Hall; 1976.

31. Rosenzweig, M. R., Bennett, E. L.; Diamond, M. C. Brain changes in response to experience. *Sci. Am., 226*, 22-29; February, 1972.

32. Rowan, A. N. *Of Mice, Models, and Men: A Critical Evaluation of Animal Research.* New York: State University of New York Press; 1984.

33. Saaty, T. L.; Alexander, J. M. *Thinking with Models.* New York: Pergamon Press; 1981.

34. Saul, A.; Showalter, K. "Propagating reaction-diffusion fronts", in *Oscillations and Traveling Waves in Chemical Systems.* Edited by R. J. Field and M. Burger; New York: Wiley; 419-439; 1985.

35. Schwarzenbach, J.; Gill, K. F. *System Modeling and Control, 2nd Edition.* London: Edward Arnold; 1978.

36. Segal, L. A. *Modeling Dynamic Phenomena in Molecular and Cellular Biology.* New York: Cambridge Press; 1984.

37. Singer, P. *Animal Liberation.* New York: Random House; 1975.

38. Stark, L. *Neurological Control Systems.* New York: Plenum Press; 1968.

39. Stent, G. S. Prematurity and uniqueness in scientific discovery. *Sci. Am., 227*: 84-93; December, 1972.

40. Thompson, W. J. *Computing in Applied Science.* New York: John Wiley & Sons; 1984.

41. Vandergraft, J. S. *Introduction to Numerical Computations, 2nd Edition.* New York: Academic Press; 1983.

42. Wolfram, S. Computer software in science and mathematics. *Sci. Am., 251*, 188-203; September, 1984.

43. Zadeh, L. A. Fuzzy sets. *Information & Control 8*: 338-353; 1965.

END
DTIC
8-86